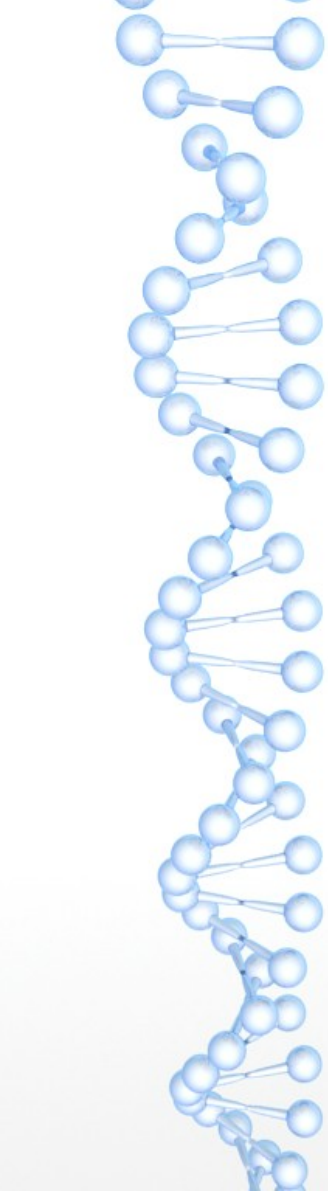


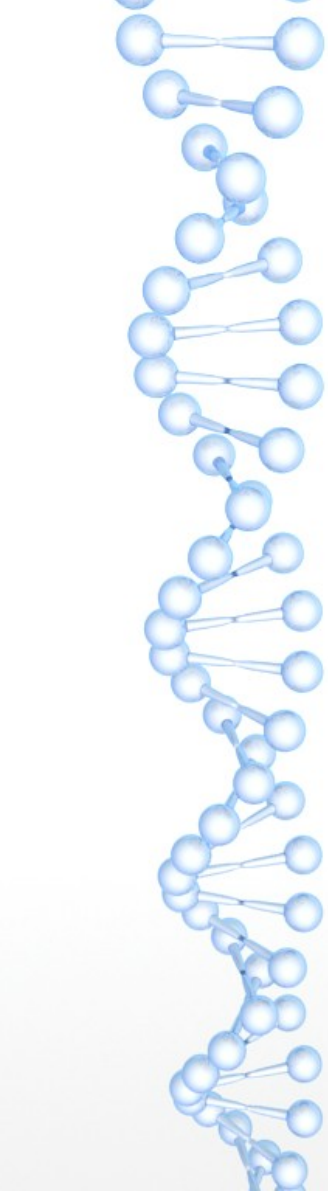
Nearest string problem

optimizacija korišćenjem RVNS i Simuliranog kaljenja

Projekat iz predmeta Računarska inteligencija

Matematički fakultet
Univerzitet u Beogradu

Sara Kapetinić 182/2017

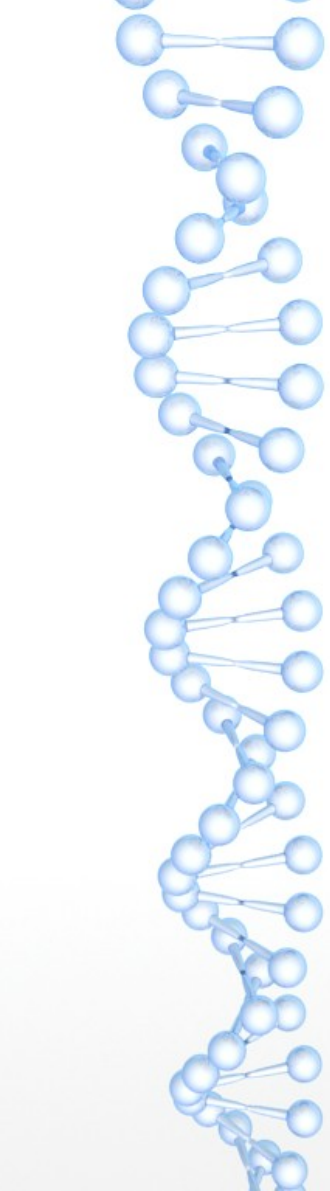


Sadržaj:

Opis problema:

- Neka je dat skup stringova $S = \{s_1, s_2, ..s_n\}$. Potrebno je pronaći string s‘ koji minimizuje $d = \max_i H(s, s_i)$, $i = 1..n$.

  H je maksimalno Hamingovo rastojanje izmedju stringa s‘ i datog skupa stringova S.
- Ovo je problem kombinatorne optimizacije i NP–težak problem.
- Nearest string problem se pojavljuje i pod drugim nazivima:
  - Closest string problem
  - Minimum Radius Problem
  - Hamming Center Problem
  - Consensus String Problem



Brute force algoritam:

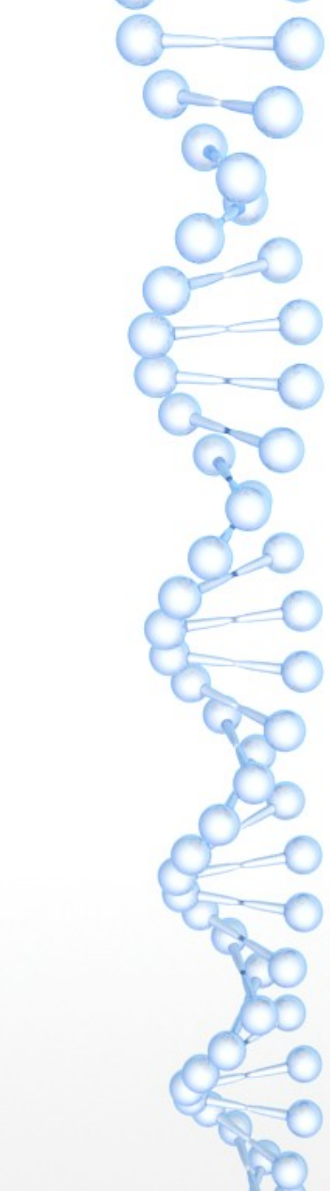
- Prvi korak u ovom algoritmu je generisanje svih permutacija nad datim alfabetom. Funkcija prima alfabet i duzinu permutacija i smesta ih u vector results.
- Funkcija findMaximumHammingDistance vraca maksimalno Hamingovo rastojanje izmedju string i skupa stringova.
- Za svaku permutaciju se pronalazi maksimalno Hamingovo rastojanje sa inicijalnim skupom stringova, a potom se bira permutacija koja ima najmanju vrednost rastojanja.
- Generisanja permutacija dosta usporava ovaj algoritam i on je poprilicno neefikasan.

RVNS algoritam:

- Reduces Variable neighborhood search – ideja je poboljsanje resenja kroz iteracije, gde krecemo od random generisanog pocetnog stringa. Proveravamo susedstva koja su zadata sa maksimalnim brojem k i pamtimo resenje ukoliko je bolje od trenutnog.
- Funkcije koje su koriscene:
  - Initialize – generise pocetno resenje
  - FindMaximumHammingDistance – vraca maksimalno Hamingovo rastojanje izmedju trenutnog resenja i skupa stringova
  - GetNeighbour – vraca k-tog suseda od trenutnog resenja(k-ti sused je onaj koji se na k mesta razlikuje od trenutnog resenja)
  - RestoreSolution – vraca prethodno resenje, pre promena

Simulirano kaljenje:

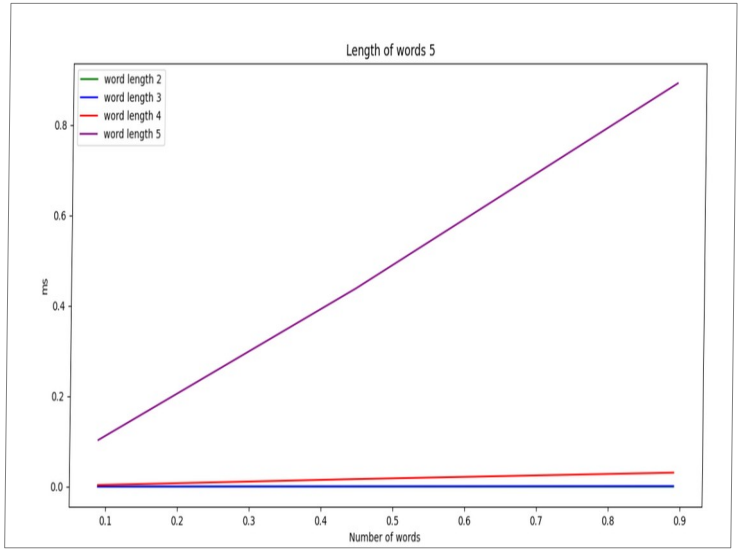
- Slican koncept kao kod prethodnog algoritma. Generisemo pocetno resenje. U svakoj iteraciji pronalazimo novo resenje kroz male izmene trenutnog. Ukoliko je izmena bila pozitivna ostavljamo novo generisano resenje. Ukoliko nije, pomocu random generatora i verovatnoce odredjujemo da li nastavljamo sa trenutnim resenjem ili ga vracamo na staro.
- Nove funkcije u odnosu na prethodni algoritam:
  - InvertSolution – menja trenutno resenje na samo jednoj poziciji
  - OldSolution – vraca staro resenje, pre promena
  - Generisanje random brojeva je iz uniformne raspodele U(0,1)



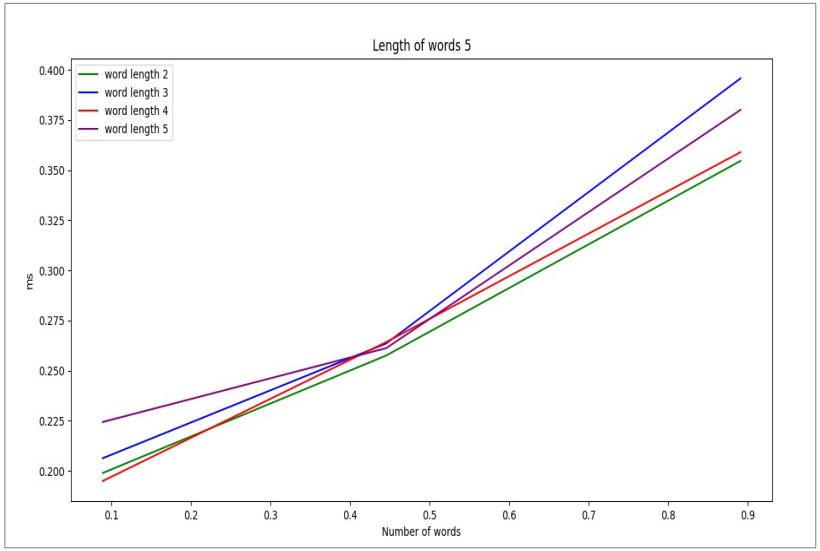
Ostale optimizacije:

- Postoji jos mnogo nacina za optimizaciju datog problema. Jedan od licnih pokusaja optimizacije ovog problema na osnovu istrazivanja istog svodi se na pamcenje broja pojavljivanja karaktera na odredjenoj poziciji u stringu.
- Radi lakseg razumevanja, postavicemo sve reci u matricu jednu ispod druge, tako da se na poziciji (i,j) nalazi j-ti karakter i-te reci.
- U svakoj koloni zapamticemo broj pojavljivanja svakog od karaktera.
- Da bismo formirali resenje problema potrebno je formirati string od karaktera koji se najcesce pojavljuju na pozicijama {1..n}, gde je n broj karaktera u resenju.

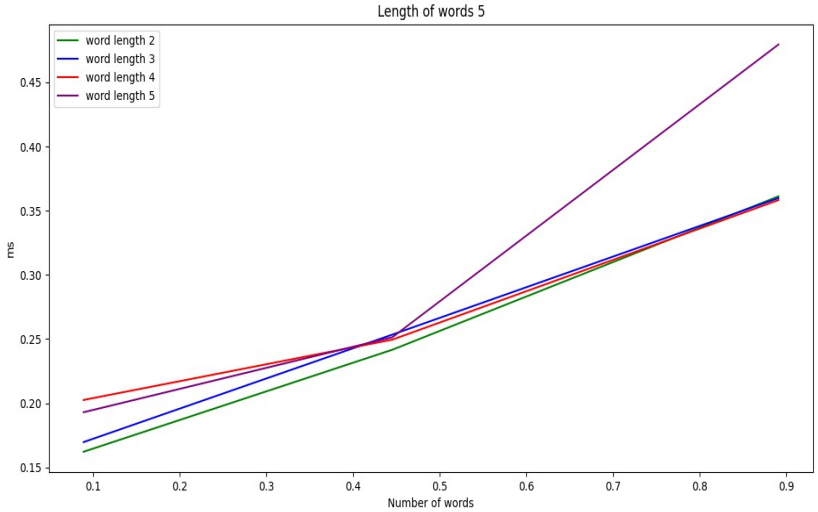
Testovi i poredjenja:



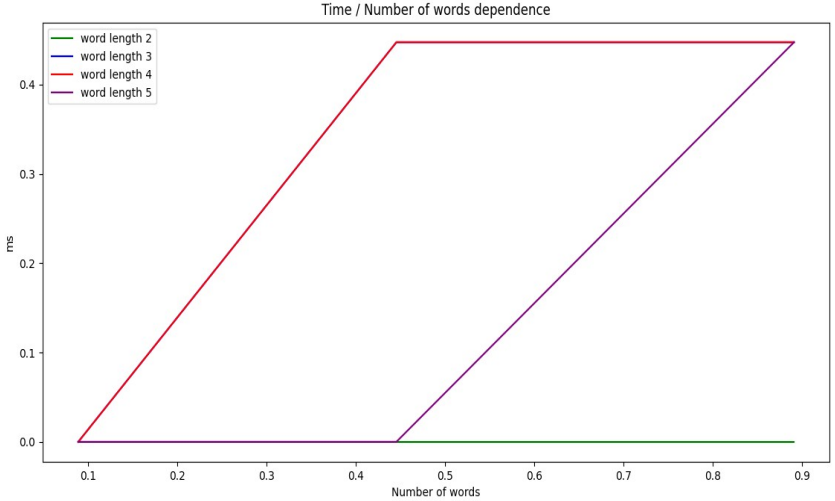
1.1 Brute force algoritam i vreme izvrsavanja programa



1.2 Simulirano kaljenje i vreme izvrsavanja za 10000 iteracija

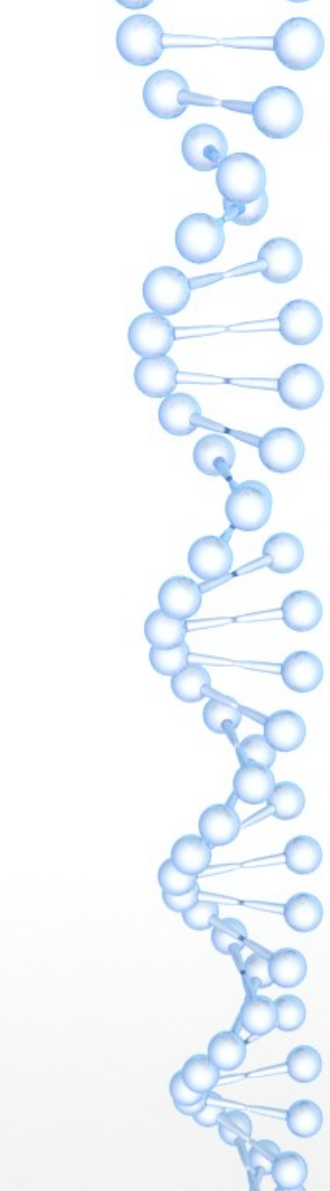


1.3 RVNS algoritam i vreme izvrsavanja



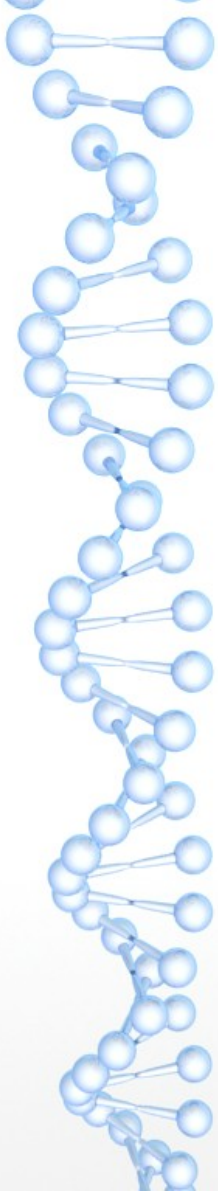
1.4 Optimizacija pamcenjem broja pojavljivanja karaktera

- Na graficima su predstavljene skalirane vrednosti. Mozemo primetiti da se na grafiku Brute force algoritma vec kod reci duzine 5 karaktera pojavljuje veliki odskok u trajanju izvrsavanja algoritma. Ostali algoritmi rade zadovoljavajuce kad je u pitanju duzina reci.
- Kod algoritama s-metaheuristika vreme izvrsavanja se povecava sa brojem iteracija programa znatno, medjutim broj reci u skupu i broj karaktera reci nisu znatno uticali na vreme izvrsavanja ekstremno kao kod brute force algoritma. Kod RVNS algoritma je duzina reci vise uticala na vreme izvrsavanja programa.



Primene:

- Nearest string je specijalan slucaj veceg problema pod nazivom Closest substring problem i obicno se srecu zajedno.
- Veliku primenu nalazimo u bioinformatici. Ovaj problem je intenzivno proucavan aspekat pronalazenja signala DNK i klasterovanju gena. Kako je poznato DNK, RNK i proteinske sekvence mogu biti poprilicno duge i optimizacija ovog problema onda postaje neophodna.



HVALA NA PAZNJI!

KRAJ